Many explicit levels

```
function memoize(f)
  let d = Dict()
    return function(args…)
      return get!(() -> f(args…), d, args)
    end
  end
end

const add = memoize(+)
const printonce = memoize(println)
```

Many explicit levels

```
macro memoize(f_expr)
  f_name, def = split_longdef(MacroTools.longdef(f_expr))
  d = Dict()
  return :($(esc(f_name))(args…) =
      get!(() -> ($(esc(def)))(args...), $d, args))
end
function split_function_def(ex)
  name = shift!(ex.args[1].args)
  ex.args[1].head = :tuple
  return name, ex
end

@memoize add(a, b) = a + b
```